Crud com Django

Básico

- Criar app
- Criar Modelos
- Criar Interface Admin
- Crir Visões
- Criar os templates

Criar APP

- python manage.py startapp <Nome da sua app> por exemplo ‘servidores’
- Depois:
  - No django_proj_name/settings.py
- INSTALLED_APPS = (

  :

  'servidores',

  :

)

Criar Modelos

- Na pasta servidores/models.py
  - from django.db import models
  - from django.core.urlresolvers import reverse
    - class Server(models.Model):
      - name = models.CharField(max_length=200)
      - ip = models.GenericIPAddressField()
      - order = models.IntegerField()

      - def __unicode__(self):
        - » return self.name
      - def get_absolute_url(self):
        - » return reverse('server_edit', kwargs={'pk': self.pk})

Criar Modelos

- Depois...é preciso criar as tabelas no banco de dados.
  - python manage.py syncdb
  - Em alguns casos é preciso rodar o migrate.

Interface de Admin

- No arquivo admin.py de servidores:
  - from django.contrib import admin
  - from servers.models import Server

  - admin.site.register(Server)

Criar as Visões

- Existem duas formas de Criar as Visões:
  - Class Based Views e Function Based Vies
- Hoje vamos ver as Class-Based Views
- https://docs.djangoproject.com/en/1.7/topics/class-based-views/

Criar as Visões

- from django.http import HttpResponse
- from django.views.generic import TemplateView,ListView
- from django.views.generic.edit import CreateView, UpdateView, DeleteView
- from django.core.urlresolvers import reverse_lazy
- from servers.models import Server

Criar as Visões

```
class ServerDelete(DeleteView):class ServerList(ListView):
    model = Server

class ServerCreate(CreateView):
    model = Server
    success_url = reverse_lazy('server_list')

class ServerUpdate(UpdateView):
    model = Server
    success_url = reverse_lazy('server_list')

class ServerDelete(DeleteView):
    model = Server
    success_url = reverse_lazy('server_list')
    model = Server
    success_url = reverse_lazy('server_list')
```

Editar o URLS.Py de Servidores

```
from django.conf.urls import patterns, url

from servers import views

urlpatterns = patterns('',
  url(r'^$', views.ServerList.as_view(), name='server_list'),
  url(r'^new$', views.ServerCreate.as_view(), name='server_new'),
  url(r'^edit/(?P<pk>\d+)$', views.ServerUpdate.as_view(), name='server_edit'),
  url(r'^delete/(?P<pk>\d+)$', views.ServerDelete.as_view(), name='server_delete'),
)
```

Editar o URLS.Py geral

```
urlpatterns = patterns('',
  :
  url(r'^servers/', include('servers.urls')),
  :
)
```

Criar Templates

- Crie o aquivo html e diretórios:
  - templates/servidores/server_form.html
- Este html vai usado para Edit e Update

```
<form method="post">{% csrf_token %}
    {{ form.as_p }}
    <input type="submit" value="Submit" />
</form>
```

Criar Templates

- Crie o aquivo html e diretórios:
  - templates/servidores/server_list.html
  - Este html vai usado para Listar os servidores

```
<h1>Servers</h1>
<ul>
   {% for server in object_list %}
   <li>{{ server.name }}  :
   <a href="{% url "server_edit" server.id
%}">{{ server.ip }}</a>
   <a href="{% url "server_delete" server.id
%}">delete</a>
   </li>
   {% endfor %}
</ul>

<a href="{% url "server_new" %}">New</a>
```

Criar Templates

- Crie o aquivo html e diretórios:
  - templates/servidores/server_confirm_delete.html
  - Este html vai usado para Apagar os servidores

```
<form method="post">{% csrf_token %}
    Are you sure you want to delete
"{{ object }}" ?
    <input type="submit" value="Submit" />
</form>
```